PENERAPAN ALGORITMA ROUND ROBIN DAN MODULO PADA LOAD BALANCING WEB SERVER

Andi Anto Diarjo [1)], Dadang Iskandar Mulyana[2)]
[1]Teknik Informatika, STIKOM CKI, [2]Teknik Informatika, STIKOM CKI
Email: andoantodiarjo@gmail.com, fokus2008@yahoo.com

Abstract: *Load balancing web server is one of the ways used to improve the performance and availability of web servers, is to divide the requests dating to multiple servers at once, so that the burden borne by each server lighter. The level of availability bias web server is maintained with the use of load balancing, namely when one server can not serve user requests (server down), then automatically the other server directly replace it, so the user as if not knowing that the server is down. The method used in load balancing is to use round robin algorithm and modulo in the server Internet Information Service. From the analysis of the implementation of load balancing has been done, the system load balancing can be one of the effective and efficient solution to create a reliable system with high availability, in particular web server.*
Keywords: *Load balancing, IIS, Round Robin, Modulo*

1. PENDAHULUAN

Dalam perkembangan teknologi saat ini, banyak orang sudah sangat membutuhkan website untuk memudahkan bisnisnya, selain itu beberapa orang sudah menyadari kebutuhan ketersedian aplikasi berbasis web untuk mempermudah manajemen dalam menjalankan bisnis proses. Saat ini ada banyak aplikasi berbasis web digunakan di instansi-instansi, dengan kebutuhan tersebut banyak orang melakukan akses ke dalam website dengan jumlah yang masif.

Dengan jumlah pengguna jaringan komputer berjumlah 1000 orang yang akan menggunakan aplikasi berbasis web maka dibutuhkan sebuah konfigurasi server yang handal. Selain sisi konfigurasi hardware yang menjadi pertimbangan agar server nanti andal maka terdapat layanan-layanan yang ada pada server harus bisa mengantisipasi pengaksesan aplikasi berbasis web tersebut secara simultan dan mempunyai frekuensi yang sangat tinggi.

Masalah akan muncul ketika penyedia jasa web hanya bergantung pada satu server tunggal saja, masalahnya antara lain overload dan crash. Masalah lainnya adalah ketersediaan server. Misalkan, pada sistem banyak prosesor akan mengalami down untuk sementara waktu jika hendak melakukan perawatan mesin atau penambahan prosesor. Hal tersebut akan mengurangi nilai pelayanan server itu sendiri.

Untuk itu akan diimplementasikan layanan load balancing yang dapat meningkatkan keandalan aplikasi berbasis web dan sistem jaringan komputer. Sebelum hal tersebut dilakukan maka infrastruktur jaringan komputer untuk lingkungan suatu instansi harus disesuaikan dengan kebutuhan.

Ada banyak metode load balancing yang digunakan sebagai penyeimbang beban server. Diantaranya adalah round robin, ratio, fastest, least connection, dan modulo. Dalam penelitian ini penulis menfokuskan pada dua jenis metode diantaranya adalah round robin dan modulo. Pada metode round robin,menggunakan algoritma yang paling sederhana dan banyak digunakan oleh perangkat load balancing. Round robin membagi beban secara bergiliran dan berurutan dari satu server ke server lain sehingga membentuk putaran. Adapun pada metode modulo, menghitung nilai mod kemudian mengirimkan permintaan ke server. Perhitungan dengan menggunakan sistem modular. Modulo untuk menyeimbangkan beban berdasarkan waktu yang dibagi, memastikan bahwa koneksi dalam sesi pengguna yang ada secara konsisten diarahkan server yang sama bahkan ketika daftar server yang tersedia diubah selama sesi pengguna.

2. METODE PENELITIAN

Dalam bentuk sederhana, operasi HTTP hanya melibatkan sebuah client HTTP, biasanya berupa browser pada computer client, dan sebuah server HTTP, lebih dikenal sebagai web server. Setelah terbentuk koneksi TCP, dua langkah komunikasi berikutnya adalah:

a. Client Request: Client HTTP mengirimkan sebuah pesan request yang terformat sesuai standar HTTP yang disebut HTTP request. Pesan ini menentukan sumber daya apa yang ingin diperoleh, atau cakupan informasi yang diberikan client kepada server

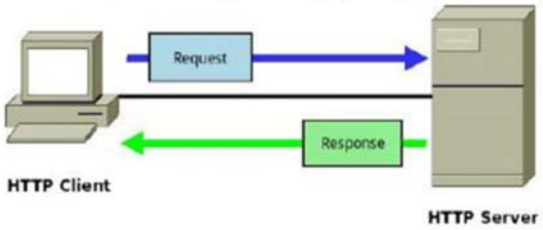
b. Server Response: Server membaca dan menterjemahkan request. Server melakukan aksi sesuai request dan mengirimkan HTTP

Response. Pesan respon ini menjadi indikator apakah request client berhasil dipenuhi dan dapat juga berisi sumber daya yang diminta oleh client.

HTTP adalah sebuah protokol meminta/menjawab antara klien dan server. Sebuah klien HTTP (seperti web browser atau robot dan lain sebagainya), biasanya

memulai permintaan dengan membuat hubungan ke port tertentu di sebuah server web hosting tertentu (biasanya port 80). Klien yang mengirimkan permintaan HTTP juga dikenal dengan user agent. Server yang meresponsnya, yang menyimpan sumber daya seperti berkas HTML dan gambar, dikenal juga sebagai origin server. Di antara user agent dan juga origin server, bisa saja ada penghubung, seperti halnya proxy, gateway, dan juga tunnel.

HTTP tidaklah terbatas untuk penggunaan dengan TCP/IP, meskipun HTTP merupakan salah satu protokol aplikasi TCP/IP paling populer melalui Internet. Memang HTTP dapat diimplementasikan di atas protokol yang lain di atas Internet atau di atas jaringan lainnya, seperti disebutkan dalam "implemented on top of any other protocol on the Internet, or on other networks.", tapi HTTP membutuhkan sebuah protokol lapisan transport yang dapat diandalkan. Protokol lainnya yang menyediakan layanan dan jaminan seperti itu juga dapat digunakan.



Gambar 2.1 Komunikasi pada HTTP

IIS atau Internet Information Services atau Internet Information Server adalah sebuah HTTP web server yang digunakan dalam sistem operasi server Windows, mulai dari Windows NT 4.0 Server, Windows 2000 Server atau Windows Server 2003. Layanan ini merupakan layanan

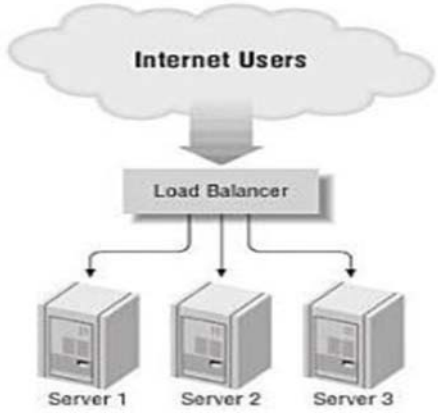
terintegrasi dalam Windows 2000 Server, Windows Server 2003 atau sebagai add-on dalam Windows NT 4.0. Layanan ini berfungsi sebagai pendukung protokol TCP/IP yang berjalan dalam lapisan aplikasi (application layer). IIS juga menjadi fondasi dari platform Internet dan Intranet Microsoft, yang mencakup Microsoft Site Server, Microsoft Commercial Internet System dan produk-produk Microsoft Back Office lainnya.

IIS telah berevolusi semenjak diperkenalkan pertama kali pada Windows NT 3.51 (meski kurang banyak digunakan) hingga IIS versi 6.0 yang terdapat dalam Windows Server 2003.Versi 5.0 diintegrasikan dalam Windows 2000, sedangkan Windows XP Professional memiliki IIS versi 5.1.Windows NT 4.0 memiliki versi 4.01 yang termasuk ke dalam add-onWindows NT Option Pack. Dalam Windows NT 4.0 Workstation atau Windows 95/98, IIS juga dapat diinstalasikan sebagai Microsoft Personal Web Server (PWS).

Webserver pertama kali yang dibuat oleh Microsoft adalah sebuah proyek riset yang dilakukan oleh sebuah lembaga yang disebut dengan European Microsoft Windows NT Academic Centre (EMWAC), bagian dari University of Edinburgh, Skotlandia dan didistribusikan sebagai perangkat lunak tak berbayar. Akan tetapi, karena memang server EMWAC tidak dapat diskalakan untuk menangani lalu lintas data yang menuju ke microsoft.com, Microsoft pun akhirnya terpaksa mengembangkan Web server miliknya sendiri, dengan nama IIS.

Load balancing adalah proses pendistribusian beban terhadap sebuah servis yang ada pada sekumpulan server atau perangkat jaringan ketika ada permintaan dari pemakai. Ketika banyak permintaan dari pemakai maka server tersebut akan terbebani karena harus melakukan proses pelayanan terhadap permintaan pemakai.

Load balancing dapat diimplementasikan dengan menggunakan perangkat keras, perangkat lunak atau gabungan keduanya. Dengan konsep yang sederhana, sebuah load balancer diletakkan di antara client dan server seperti terlihat pada Gambar 2.0. Load balancer akan menampung lalu lintas yang datang dan membaginya ke dalam request-request individual lalu menentukan server mana yang menerima request tersebut.



Gambar 2.2 Penerapan Load Balancing

Beberapa keuntungan dari penerapan load balancing antara lain :

a. Scalability: Ketika beban sistem meningkat, kita dapat melakukan perubahan terhadap sistem agar dapat mengatasi beban sesuai dengan kebutuhan.
b. High Availability: Load balancer secara terus menerus melakukan pemantauan terhadap server. Jika terdapat server yang mati, maka load balancer akan menghentikan request ke server tersebut dan mengalihkannya ke server yang lain.
c. Manageability: Mudah ditata meskipun secara fisik sistem yang ditanam sangat besar.
d. Security: Untuk semua lalu lintas yang melewati

load balancer, aturankeamanan dapat diimplementasikan dengan mudah. Dengan private network digunakan untuk server, alamat IP nya tidak akan diakses secara langsung dari luar sistem.

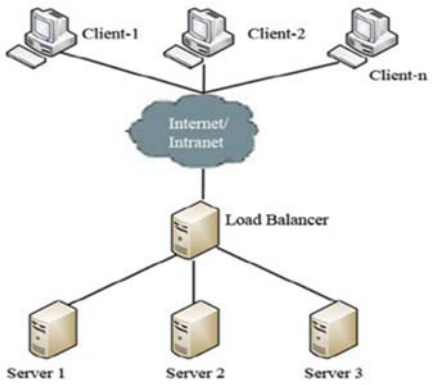
Beberapa jenis algoritma penjadwalan yang dapat diterapkan pada sistem web server IIS pada proses distribusi request kepada real server, antara lain yaitu:

a. Round Robin (rr), yaitu algoritma penjadwalan yang memperlakukan semua real server sama menurut jumlah koneksi atau waktu respon.
b. Weighted Round Robin (wrr), penjadwalan ini memperlakukan real server dengan kapasitas proses yang berbeda. Masing-masing real server dapat diberi bobot bilangan integer yang menunjukkan kapasitas proses, dimana bobot awal adalah 1.
c. Least Connection (lc), merupakan algoritma penjadwalan yang mengarahkan koneksi jaringan pada server aktif dengan jumlah koneksi yang paling sedikit. Penjadwalan ini termasuk salah satu algoritma penjadwalan dinamik, karena memerlukan perhitungan koneksi aktif untuk masing-masing real server secara dinamik. Metode penjadwalan ini baik digunakan untuk melancarkan pendistribusian ketika request yang datang banyak.
d. Weighted Least Connection (wlc), merupakan sekumpulan penjadwalan least connection dimana dapat ditentukan bobot kinerja pada masing-masing real server. Server dengan nilai bobot yang lebih tinggi akan menerima persentase yang lebih besar dari koneksi-koneksi aktif pada satuwaktu. Bobot pada masing-masingreal server dapat ditentukan dan koneksi jaringan dijadwalkan pada masing-masing real server dengan persentase jumlah koneksi aktif untuk masing-masing server sesuai dengan perbandingan bobotnya (bobot awal adalah 1).
e. Locality Based Least Connection (lblc), metode penjadwalan yang akan mendistribusikan lebih banyak request kepada real server yang memiliki koneksi kurang aktif. Algoritma ini akan meneruskan semua request kepada real server yang memiliki koneksi kurang aktif tersebut sampai kapasitasnya terpenuhi.
f. Hashing (h), merupakan algoritma penjadwalan statik yang dapat meneruskan request dari client kepada satu real server tertentu sesuai dengan layanan yang diminta. Terdapat suatu tabel hash berisi alamat tujuan dari masing-masing real server beserta layanan yang tersedia pada setiap real server. Atau dengan sistem pembagian dengan script modulo.

Metode Pembuktian Simulasi

1. Arsitektur Sistem

Pengujian pada penelitian ini meggunakan satu aplikasi yang akan di load secara terus menerus. Untuk mechine yang disimulasikan menggunakan satu komputer yang dibuat seperti menggunakan tiga server. Session-session komunikasi dibuka oleh server untuk memungkinkan para penggunanya menikmati servis dari server. Jika satu server saja yang dibebani, tentu server tersebut tidak akan dapat melayani banyak pengguna, karena kemampuannya dalam melakukan processing ada batasnya. Batasan ini bias berasal dari banyak hal, misalnya kemampuan processing-nya, bandwidth internetnya, dan banyak lagi.



Gambar 2.3 Simulasi Load Balancing

Untuk itu, solusi yang paling ideal adalah dengan membagi-bagi beban yang dating tersebut ke beberapa server, sesuai dengan gambar 3.1. Jadi, yang bertugas melayani pengguna tidak hanya terpusat pada satu perangkat saja. Inilah yang disebut system load balancing.

Misalnya ketika kita mengakses ke situs www.website.com, maka web server yang berisi dokumen-dokumen berita, akan langsung melayani kita. Server tersebut memberikan apa yang kita minta dengan membuka komunikasi menggunakan servis HTTP port 80. Informasi halaman utama akan langsung dikirimkan ke PC melalui port 80 tersebut, sehingga kita dapat melihatnya di halaman browser.

Ketika kita melakukan akses suatu link pada halaman web tersebut, permintaan kita kemudian diproses kembali oleh server. Web server akan melayani permintaan kita lagi dengan berbagai cara yang telah ditentukan oleh pengelolanya, apakah mengarahkan kita ke dalam folder tertentu, menjalankan script-script tertentu, mengirimkan gambar, memutar klip suara, dan banyak lagi. Pada saat ini, server www.website.com sedang terbebani oleh permintaan kita. Hingga halaman atau layanan yang kita minta terbuka, maka selesailah proses tersebut dan server kembali bebas dari beban.

Jika yang mengakses halaman web www.website.com hanya kita seorang, tentu system load balancing tidak diperlukan, karena sebuah server tentu masih sangat cukup untuk melayani permintaan kita. Namun apa jadinya jika www.website.com dibuka oleh hampir sebagian besar pengguna internet di

Indonesia, setiap detik, dan setiap hari seperti keadaan saat ini. Mungkin sebuah server saja tidak akan sanggup melayani permintaan seberat itu. Permintaan akan terus dating dan proses juga akan terus-menerus dilakukan.

2. Cara Kerja Sistem.

Sistem yang akan diuji adalah halaman load runner sebagai single server yang akan di load oleh para user. Pembagian dari permintaan (load) dengan membuat tiga website di dalam IIS dengan dibedakan port protocol HTTP, dengan rincian sesuai dengan tabel 2.0

Tabel 2.0 Pembagian port dalam Web Server

Website	*Port*
Website 1	80
Website 2	81
Website 3	83

Ketiga website tersebut memiliki 1 aplikasi yaitu load runner. Pembagian tersebut menggunakan aplikasi load mapper. Di dalam halaman load mapper adalah inti pengujian, sesuai dengan gambar 3.1, yang menjadi load balancer pada pengujian ini adalah aplikasi load mapper. Cara kerja sistem yang akan diuji dengan menggunakan algoritma round robin dan algoritma modulo. Berikut ini adalah elaborasi dari metode tersebut.

a. Round Robin

Konsep dasar dari algoritma ini adalah dengan menggunakan time-sharing. Pada dasarnya algoritma ini sama dengan FCFS (First-Come First-Serve), hanya saja bersifat preemptive. Setiap proses mendapatkan waktu mesin yang disebut dengan waktu quantum (quantum time) untuk membatasi waktu proses, biasanya 1-100 milidetik. Setelah waktu habis, proses ditunda dan ditambahkan pada ready queue.

Jika suatu proses memiliki burst time lebih kecil dibandingkan dengan waktu quantum, maka proses tersebut akan melepaskan server jika telah selesai bekerja, sehingga server dapat segera digunakan oleh proses selanjutnya. Sebaliknya, jika suatu proses memiliki bursttime yang lebih besar dibandingkan dengan waktu quantum, maka proses tersebutakan dihentikan sementara jika sudah mencapai waktu quantum, dan selanjutnya mengantri kembali pada posisi ekor dari ready queue, server kemudian menjalankan proses berikutnya.

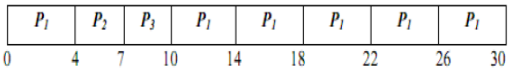
Jika terdapat n proses pada ready queue dan waktu quantum q, maka setiap proses mendapatkan 1/n dari waktu server paling banyak q unit waktu pada sekali penjadwalan server. Tidak ada proses yang menunggu lebih dari (n-1) * q unit waktu. Performansi algoritma round robin dapat dijelaskan sebagai berikut, jika q besar, maka yang digunakan adalah algoritma FIFO (First In First Out), tetapi jika q kecil maka sering terjadi context switch.

Misalkan ada 3 proses: P1, P2, dan P3 yang meminta pelayanan server dengan quantum-time sebesar 4 milidetik.

Tabel 2.1 Uji coba Burst Time

Process	*Burst Time*
P1	24
P2	3
P3	3

Penjadwalan proses dengan algoritma round robin dapat dilihat pada gant chart berikut:

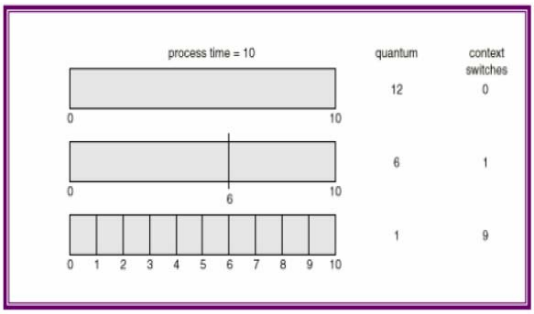


Gambar 2.4 Gant Chart Round Robin

Waktu tunggu untuk P1 adalah 6, P2 adalah 4, dan P3 adalah 7 sehingga rata-rata waktu tunggu adalah (6 + 4 + 7)/3 = 5.66 milidetik.

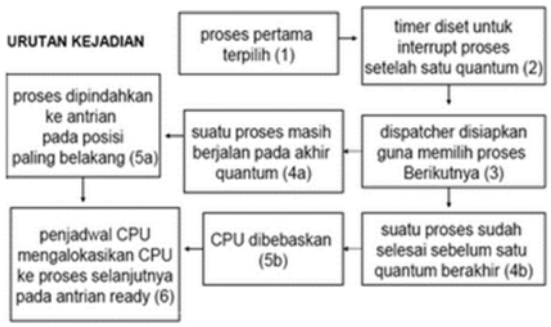
Algoritma round robin ini di satu sisi memiliki keuntungan, yaitu adanya keseragaman waktu. Namun permasalahan utama pada Round Robin adalah menentukan besarnya time quantum. Jika time quantum

yang ditentukan terlalu kecil, maka sebagian besar proses tidak akan selesai dalam 1 quantum. Hal ini tidak baik karena akan terjadi banyak switching, padahal server memerlukan waktu untuk beralih dari suatu proses ke proses lain (disebut dengan context switches time). Sebaliknya, jika time quantum terlalu besar, algoritma Round Robin akan berjalan seperti algoritma first come first served. Time quantum yang ideal adalah jika 80% dari total proses memiliki burst time yang lebih kecil dari 1 quantum time. Seperti yang terlihat pada Gambar 2.5. Semakin besar quantum time maka switching yang terjadi akan semakin sedikit.



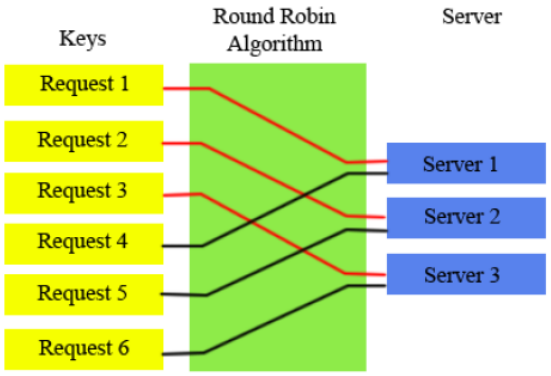
Gambar 2.5 Waktu quantum meningkatkan context switch

Sesuai dengan penjelasan diatas algoritma penjadwalan round robin memiliki urutan kejadian yang dijelaskan pada gambar 2.5



Gambar 2.6 Urutan kejadian round robin

Pada pengujian round robin akan dibuat variable nextTarget kemudian akan di cek variable tersebut apakah null atau tidak. Jika bernilai null maka akan di kirim ke halaman http://localhost:80/loadrunner (Server 1), dan varible nextTarget akan diisi dengan link tersebut. Setelah itu akan di cek variable nextTarget, bila memiliki nilai http://localhost:80/loadrunner (Server 1), maka akan dikirim ke halaman http://localhost:81/loadrunner (Server 2), dan variable nextTarget akan diisi dengan nilai link tersebut. Kemudian variable nextTarget akan dicek kembali, bila memiliki nilai http://localhost:81/loadrunner (Server 2), maka akan dikirim ke halaman http://localhost:83/loadrunner (Server 3) dan variable nextTarget akan diisi dengan nilai link tersebut. Kemudian nextTarget akan dicek kembali dan bila memiliki nilai http://localhost:83/loadrunner (Server 3) maka akan dikirim ke halaman http://localhost:80/loadrunner (Server 1) dan nextTarget akan diisi dengan nilai link tersebut.



Gambar 2.7 Round Robin Algorithm

Untuk mengetahui hasil pengujian diatas berikut adalah tabel hasil capture dari lagoritma round robin.

Tabel 2.2 Pemetaan server dengan algoritma round robin

Pemetaan	Percobaan
SERVER 1 = http://localhost/loadrunner/default.aspx 2011/08/09 22:22:53	1
SERVER 2 = http://localhost:81/loadrunner/default.aspx 2011/08/09 22:25:43	2
SERVER 3 = http://localhost:83/loadrunner/default.aspx 2011/08/09 22:26:42	3
SERVER 1 = http://localhost/loadrunner/default.aspx 2011/08/09 22:29:03	4
SERVER 2 = http://localhost:81/loadrunner/default.aspx 2011/08/09 22:30:22	5
SERVER 3 = http://localhost:83/loadrunner/default.aspx 2011/08/09 22:31:02	6

Pada tabel 2.2 diatas menunjukkan dari 6 percobaan request yang dilakukan, pemetaan server dengan menggunakan algoritma round robin sangat rapi. Request pertama akan diarahkan ke server 1, kemudian request kedua akan diarahkan ke server 2, dan request ketiga diarahkan ke server 3, untuk request keempat akan diarahkan kembali ke server 1.

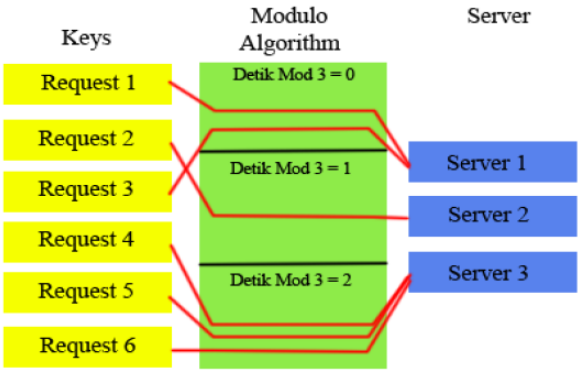
b. Modulo-Based

Dalam matematika dan pemrograman komputer, operasi modulus adalah sebuah operasi yang menghasilkan sisa pembagian dari suatu bilangan terhadap bilangan lainnya. Dalam bahasa pemrograman operasi ini umumnya dilambangkan dengan simbol %, mod atau modulo, tergantung bahasa pemrograman yang digunakan. Misalkan dua bilangan a dan b, a modulo b (disingkat a mod b) adalah bilangan bulat sisa pembagian a oleh b. Misalnya, "1 mod 3", "4 mod 3", dan "7 mod 3" memiliki hasil 1, karena ketiga bilangan tersebut memiliki sisa 1 jika dibagi oleh 3, sedangkan "9 mod 3" sama dengan 0. Penerapan operasi modulus dalam teori bilangan tergolong kepada aritmatika modulus.

Dalam cara ini eksekusi modulo bekerja berdasarkan waktu pengaksesan aplikasi dengan algoritma waktu yang sekarang (date now) mod 3. Dalam penelitian ini akan disimulasikan dengan dua pendekatan yaitu waktu sekarang dengan satuan detik mod 3 dan dan waktu sekarang dengan satuan menit mod 3. Pada perhitungan detik, Jika kita masuk pada detik yang dibagi 3 memiliki sisa 0 maka kita akan dikirim ke website dengan port 80 yang memiliki halaman website http://localhost:80/loadrunner (Server 1). Dan ketika kita masuk pada detik dibagi 3 yang memiliki sisa 1 maka kita akan dikirim ke website dengan port 81 yang memiliki halaman website

http://localhost:81/loadrunner (Server 2). Dan jika kita masuk pada detik dibagi 3 yang memiliki sisa 0 maka akan dikirim pada port 83 yang memiliki halaman website http://localhost:83/loadrunner (Server 3).

Pada perhitungan menit, Jika kita masuk pada menit yang dibagi 3 memiliki sisa 0 maka kita akan dikirim ke website dengan port 80 yang memiliki halaman website http://localhost:80/loadrunner (Server 1). Dan ketika kita masuk pada menit dibagi 3 yang memiliki sisa 1 maka kita akan dikirim ke website dengan port 81 yang memiliki halaman website http://localhost:81/loadrunner (Server 2). Dan jika kita masuk pada menit dibagi 3 yang memiliki sisa 0 maka akan dikirim pada port 83 yang memiliki halaman website http://localhost:83/loadrunner (Server 3).



Gambar 2.8 Detik Modulo Algorithm

Untuk mengetahui hasil pengujian diatas berikut adalah tabel hasil capture dari lagoritma modulo.

Tabel 2.3 Pemetaan server dengan algoritma modulo

Pemetaan	Percobaan
SERVER 1 = http://localhost/loadrunner/default.aspx 2011/08/09 20:27:03	1
SERVER 3 = http://localhost:83/loadrunner/default.aspx 2011/08/09 20:27:56	2
SERVER 2 = http://localhost:81/loadrunner/default.aspx 2011/08/09 20:28:40	3

Terlihat dari tabel 2.3 diatas bahwa percobaan ini yang diambil adalah detik pengaksesan yang akan dibagi tiga dengan algoritma modulo. Pada percobaan pertama menunjukkan detik saat mengkases server adalah 03, maka perhitungaannya adalah:

3 mod 3 = 0 □ Sisa 0

Karena memiliki sisa 0 maka server yang akan diakses adalah server 1. Sedangkan pada percobaan kedua waktu pengaksesan penunjukkan pada detik ke 56, sehingga perhitungannya adalah:

56 mod 3 = ((3*18)+2) mod 3 = 2 mod 3 = 2 □ Sisa 2

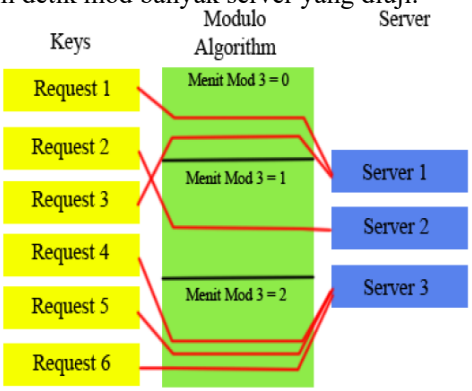
Karena memiliki sisa 2 maka server yang akan diakses adalah server 3. Untuk percobaan terakhir waktu pengaksesan penunjukkan pada detik ke 40, sehingga perhitungannya adalah:

40 mod 3 = ((3*13)+1) mod 3 = 1 mod 3 = 1 □ sisa 1

Karena memiliki sisa 1 maka server yang akan diakses adalah server 1.

Selain dalam detik kita juga menguji dalam Menit. Pada perhitungan menit, jika kita masuk pada menit yang dibagi 3 memiliki sisa 0 maka kita akan dikirim ke website dengan port 80 yang memiliki halaman websitehttp://localhost:80/loadrunner (Server 1). Dan ketika kita masuk pada menit dibagi 3 yang memiliki sisa 1 maka kita akan dikirim ke website dengan port 81 yang memiliki halaman websitehttp://localhost:81/loadrunner (Server 2). Dan jika kita masuk pada menit dibagi 3 yang memiliki sisa 0 maka akan dikirim pada port 83 yang memiliki halaman websitehttp://localhost:83/loadrunner (Server 3). Simulasi perhitungan menit algoritma modulo dapat dilihat pada gambar 2.9.

Dalam eksperimen hanya dilakukan perhitungan dalam detik mod banyak server yang diuji.



Gambar 2.9 Menit ModuloAlgorithm

3. Skenario Eksperimen

Eksperimen yang dilakukan dalam penelitian ini bertujuan untuk melakukan pengujian terhadap algoritma round robin dan algoritma modulo yang diusulkan. Skenario pengujian eksperimen menggunakan tool yang bernama WAPT (Web Application Testing) yang berfungsi sebagai pengujian beban yang berada pada situs web, selain itu tool ini dapat menghasilakan grafik dari eksperimen yang dilakukan. Selain grafik, tool juga memiliki data statistik pengaksesan ketika server diberikan beban yang berat. Tool ini sering digunakan untuk menguji aplikasi bisnis pribadi yang digunakan untuk web portal terdistribusi terdiri dari load balancer, web server, application server, dan database storages, eksperimen dilakukan dengan menggunakan dua skenario, yang pertama adalah Eksperiment Response Time dan Eksperiment Balancing Strategy.

Skenario Eksperiment Stress Test

Skenario ini bertujuan untuk mengetahui rata- rata halaman yang dieksekusi per detik dan mengetahui rata-rata waktu respon jika aplikasi load balancer di berikan tekanan lebih dengan simulasi meggunakan WAPT.

Pengujian dengan membandingkan perbedaan rata-rata halaman yang dieksekusi per detik dan waktu respon dari single server, server yang memiliki algorima load balancing round robin dan server yang memiliki algoritma load balancing modulo. Parbandingan dapat dilihat dari grafik response time, rata-rata response time, dan overall performance. Ketika mendapatkan data-data tersebut akan terlihat perbandingan ketiganya. b. Skenario Eksperiment Balancing Strategy.

Skenario ini bertujuan untuk mengetahui pemetaan request dengan menggunakan teknik berbeda. Yaitu dengan algoritma round robin dan algoritma modulo. Dengan dua teknik ini kita akan membandingkan jumlah request yang masuk kedalam masing-masing server. Dengan jumlah request tersebut kita dapat menganalisa tingkat kestabilan dari server-server yang disediakan tersebut. Selain tingkat kestablan juga dapat melihat kesempurnaan pemetaan dari dua algoritma yang digunakan.

3. HASIL DAN PEMBAHASAN

Persiapan dilakukan dengan konfigurasi web server (Internet Information Services), dengan memilih Application Pool, Port yang digunakan dalam pengujian ini adalah port 80, port 81, dan port 83.

Langkah selanjutnya dalam pengujian adalah membuat halaman load runner. Fungsi dari halaman ini sebagai halaman tujuan dari pengujian aplikasi, atau halaman yang akan diakses oleh user ketika melakukan melakukan load, halaman ini juga yang akan di tanamkan ke dalam web server yang berisi load balancer. Pembuatan script menggunakan bahasa pemrograman C#.Net. Halaman ini berisi label yang akan diisi dengan detik pengaksesan.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Load Runner</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:ScriptManager ID="ScriptManager1" runat="server" />
        <div>
            <asp:Label ID="lblCounter" runat="server" Text="Label"></asp:Label>
            <asp:Timer ID="tmrLoad" runat="server" Interval="3000" OnTick="tmrLoad_Tick">
            </asp:Timer>
        </div>
    </form>
</body>
</html>
```

Label diatas diisi dengan script yang menampilkan lokasi real url yang telah diakses oleh user, selain itu penambahan isi string juga diisikan date now atau waktu pada saat melakukan load kedalam aplikasi ini dengan format tahun/bulan/ tanggal dan jam:menit:detik. berbeda. Yaitu dengan algoritma round robin dan algoritma modulo. Dengan dua teknik ini kita akan membandingkan jumlah request yang masuk kedalam masing-masing server. Dengan jumlah request tersebut kita dapat menganalisa tingkat kestabilan dari server-server yang disediakan tersebut. Selain tingkat kestablan juga dapat melihat kesempurnaan pemetaan dari dua algoritma yang digunakan.

```
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
            lblCounter.Text = Request.Url.AbsoluteUri + " " + DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss");
    }
}
```

Load runner di bagi-bagi oleh tiga server yang dibuat pada langkah sebelumnya dengan pembagian port 80, port 81, dan port 83.

Setelah membuat halaman load runner, langkah selanjutnya adalah membuat halaman load mapping. Halaman ini adalah inti dari pengujian yang bisa disebut dengan load balancer. Pada halaman ini algoritma round robin dan algoritma modulo dibuat sebagai pembagi beban pada server.

Pada pengujian round robin akan dibuat variable nextTarget kemudian akan di cek variable tersebut apakah null atau tidak. Jika bernilai null maka akan dibuat variabel counter yang akan diisi dengan nilai 1 sebagai penghitung berapa sering Server 1 dikunjungi. Kemudian load akan dikirim ke halaman http://localhost:80/loadrunner (Server1), dan varible nextTarget akan diisi dengan linkhttp://localhost:81/loadrunner.

```
if (Application["NextTarget"] == null)
{
    Application["counter1"] = 1;
    Application["NextTarget"] = "http://localhost:81/loadrunner";
    Response.Redirect("http://localhost:80/loadrunner", false);
}
```

Jika variable nextTarget tidak bernilai null maka akan dicek isi dari variable nextTarget. Jika nextTarget bernilai http://localhost:81/loadrunner(Server 2), maka akan diisi varibel counter 2 dengan nilai1, dan akan diperbaharui nilai dari variable nextTarget adalah http://localhost:83/loadrunner(Server 3), setelah itu akan dikirim kehalaman http://localhost:81/loadrunner(Server 2).

```
else if (currentTarget == "http://localhost:81/loadrunner")
{
    if (Application["counter2"] == null)
        Application["counter2"] = 1;
    else
        Application["counter2"] = (int)Application["counter2"] + 1;

    Application["NextTarget"] = "http://localhost:83/loadrunner";
    Response.Redirect(currentTarget, false);
}
```

Kemudian variable nextTarget akan dicek kembali. Jika nextTarget bernilai http://localhost:83/loadrunner (Server 3), maka akan diisi variabel counter 3 dengan nilai 1, dan akan diperbaharui nilai dari variable next

Target adalah http://localhost:80/loadrunner(Server 1), setelah itu akan dikirim kehalaman http://localhost:83/loadrunner(Server 3).

```
else if (currentTarget == "http://localhost:83/loadrunner")
{
    if (Application["counter3"] == null)
        Application["counter3"] = 1;
    else
        Application["counter3"] = (int)Application["counter3"] + 1;

    Application["NextTarget"] = "http://localhost:80/loadrunner";
    Response.Redirect(currentTarget, false);
}
```

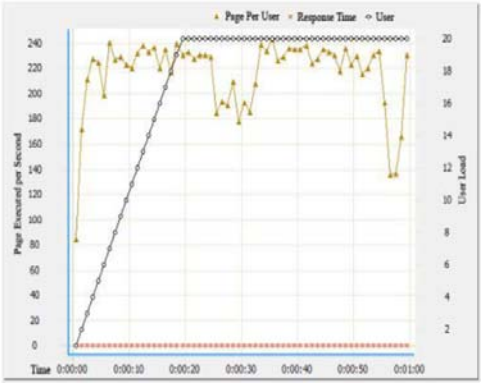
Catatannya adalah variabel counter akan terus di perbaharui dengan ditambah 1 setiap ada permitaan ke masing-masing server yang di load oleh user.

Analisis Skenario Eksperiment Stress Test

Peda pengujian ini aplikasi akan diberikan tekanan dengan menggunakan aplikasi WAPT. Bertujuan untuk mengetahui rata-rata halaman yang dieksekusi setiap detik dan mengetahui rata-rata waktu respon ketika di load.

Pada pengujian ini kita akan membandingkan perbedaan rata-rata (average) waktu respon yang dibutuhkan dari website dalam melakukan load ke dalam server. Pengujian yang yang dilakukan yaitu mambadingkan waktu respon yang terdiri dari Single Server, Load Balancing Round Robin, Load Balancing Modulo Base.

Eksperimen dilakukan dengan skenario menggunakan user 1-20. Dengan ketentuan step perubahan user ketika melakukan load ke dalam server dengan step penambahan user 1 user per satu detik, dan durasi pengetesan simulasi load adalah sepanjang 60 Detik. Berikut ini akan dielaborasi dari semua kemungkinan perbandingan. Berikut adalah gambar 3.0 yang menunjukkan hasil pengujian stress test dengan menggunakan single server.



Gambar 3.0 Grafik Pengujian Single Server

Pada grafik diatas terlihat sumbu y sebelah kiri menunjukkan halaman yang dieksekusi per detik, sumbu y sebelah kanan menunjukkan pengguna yang melakukan load ke dalam aplikasi, dan pada sumbu x menunjukkan waktu pengujian. Untuk tanda segitiga warna coklat adalah jumlah halaman yang diakses oleh user, untuk lingkaran warna hitam adalah jumlah user yang melakukan akses ke dalam aplikasi, dan kotak warna merah adalah waktu respon.

Terlihat pada grafik pengujian single server halaman yang bisa dieksekusi oleh user yang paling tinggi adalah 240 halaman yang dieksekusi, untuk rata-rata halaman yang yang dieksekusi per detik adalah 637. Setiap detiknya akan bertambah 1 user yang mengeksekusi sehingga pada detik yang ke 20 dan seterusnya akan sama jumlah semua user yang melakukan load. Dan waktu respon setiap detik cukup stabil, untuk detail response time dapat dilihat pada lampiran A1, average response time untuk single serve adalah 0,02 detik. Berikut ini adalah grafik load balancing round robin.

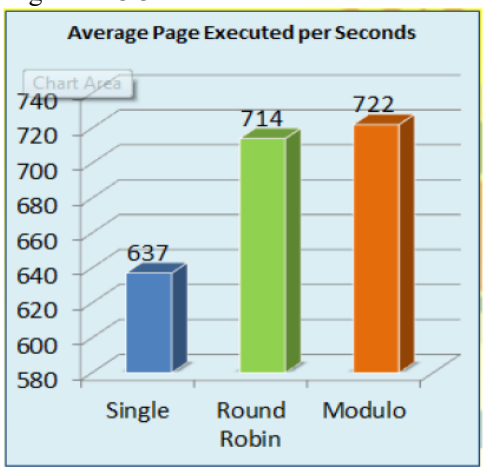
Gambar 3.1 Grafik Pengujian Load Balancing Round Robin

Terlihat pada grafik pengujian multi server dengan load balancing round robin halaman yang bisa dieksekusi oleh user yang paling tinggi adalah 800 halaman yang dieksekusi,. untuk rata-rata halaman yang yang dieksekusi per detik adalah 714. Setiap detiknya akan bertambah 1 user yang mengeksekusi sehingga pada detik yang ke 20 dan seterusnya akan sama jumlah semua user yang melakukan load. Dan waktu respon setiap detik cukup stabil, untuk detail response time dapat dilihat pada lampiran A2, terlihat pada lampiran A2 bahwa pengujian ini user melakukan load ke dua halaman yaitu halaman load runner dan halamana load mapper. Pada pengaksesan load mapper dibutuhkan waktu 0,009 dan pengaksesan halaman load mapper adalah 0,03 sehingga rata-rata response time untuk load balancing round robin adalah adalah 0,0195 detik.

Gambar 3.2 Grafik Pengujian Load Balancing Modulo-based

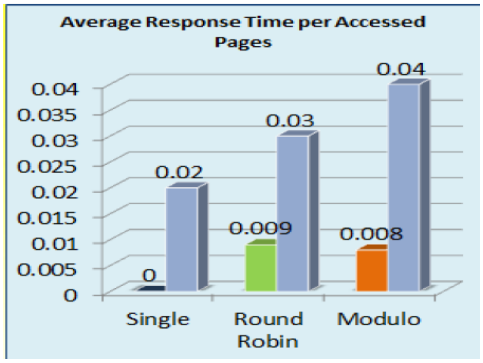
Terlihat pada grafik pengujian multi server dengan load balancing modulo based halaman yang bisa dieksekusi oleh user yang paling tinggi adalah 800 halaman yang dieksekusi, untuk rata-rata halaman yang yang dieksekusi per detik adalah 722. Dan setiap detiknya akan bertambah 1 user yang mengeksekusi sehingga pada detik yang ke 20 dan seterusnya akan sama jumlah semua user yang melakukan load. Dan waktu respon setiap detik cukup stabil, untuk detail response time dapat dilihat pada lampiran A3, terlihat pada lampiran A3 bahwa pengujian ini user melakukan load ke dua halaman yaitu halaman load runner dan halamana load mapper. Pada pengaksesan load mapper dibutuhkan waktu 0,008 dan pengaksesan halaman load mapper adalah 0,04 sehingga rata-rata response time untuk load balancing round robin adalah adalah 0,024 detik.

Sehingga dari ketiga pengujian di atas dapat dibandingkan berdasarankan halaman yang dieksekusi per detik gambar 3.3 di bawah ini.

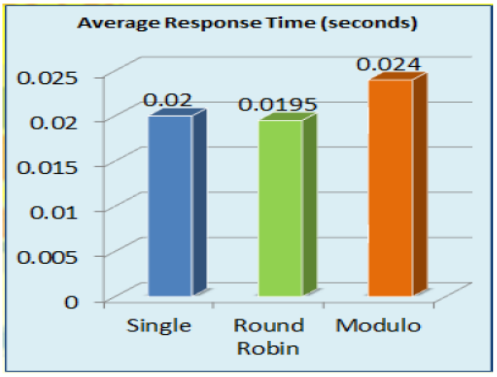


Gambar 3.3 Grafik Perbandingan Average Page Execute Per seconds

Halaman yang dieksekusi perhalaman untuk multi server dengan load balancing modulo memiliki nilai rata-rata paling tinggi yaitu 722 halaman yang dieksekusi. Sedangkan untuk multi server dengan load balancing round robin memiliki nilai rata-rata 714. Dan untuk single server memiliki rata-rata 637 halaman yang dieksekusi. Dari percobaan eksperimen ini multi server dengan menggunakan load balancing memiliki rata-rata halaman yang dieksekusi perdetik libih baik dari pada single server. Untuk pengujian response time dapat dilihat pada gambar di bawah ini.



Gambar 3.4 Grafik Average Response Time per Accessed Page



Gambar 3.5 Grafik Average Response Time

Pada gambar 3.4 menunjukkan bahwa rata-rata waktu respon ketika melakukan akses ke halaman. Untuk single server tidak ada pengaksesan pertama karena langsung diarahkan ke halaman utama dan rata-ratanya adalah 0,02 detik. Untuk load balancing round robin melakukan dua akses, halaman pertama yang diases adalah halaman load mapper dengan rata-rata 0,009 detik dan halaman yang ke dua memiliki rata-rata 0,03 detik. Sedangakan untuk rata-ratanya adalah 0,0195 detik. Untuk load balancing modulo-based melakukan dua akses, halaman pertama yang diases adalah halaman load mapper dengan rata-rata 0,008 detik dan halaman yang ke dua memiliki rata-rata 0,04 detik. Sedangakan untuk rata-ratanya adalah 0,024 detik. Sehingga secara keseluruhan load balancing dengan teknik round robin memiliki waktu respon yang cukup cepat.

Analisis Skenario Eksperiment Balancing Strategy

Pada Skenario ini kita akan membandingkan pemetaan dengan teknik berbeda, yaitu dengan menggunakan metode algoritma Round Robin dan

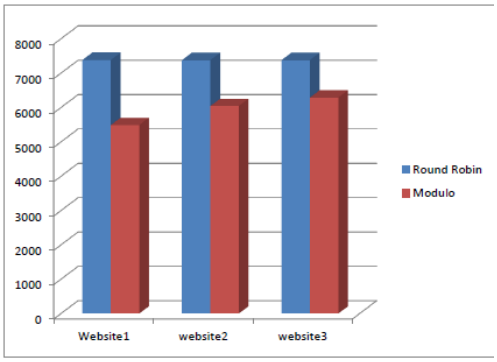
algoritma Modulo Base.

Eksperimen dilakukan dengan skenario menggunakan user 1-20. Dengan ketentuan step perubahan user ketika melakukan load ke dalam server dengan step penambahan satu user per satu detik, dan durasi pengetesan simulasi load adalah sepanjang 60 Detik. Pengujian ini akan membanding besaran request yang masuk kedalam server dari 2 teknik tersebut.

Untuk pengujian eksperimen balancing strategy juga menggunakan menggunakan software Web Application Web (WAPT) memiliki data yang sama dengan eksperimen stress test. berikut ini merupakan table 3.0 adalah jumlah request ke server.

Tabel 3.0 Jumlah Request ke Server

Device	*Round Robin*	*Modulo*
Server1	7381	5488
Server2	7378	6042
Server3	7378	6287
Total	22137	17817



Gambar 3.6 Grafik Jumlah Request ke Server

Dari table 3.0 menunjukkan jumlah request dari round robin lebih besar dari pada modulo base, ini menunjukkan bahwa round robin memiliki pemetaan dan pembagian load ke server lebih baik dari modulo base. Dari gambar 3.3 juga terlihat stabilitas dari round robin lebih baik dari pada modulo base.

Evaluasi Kinerja Sistem

1. Evaluasi Analisis Skenario Eksperiment Stress Test Dari eksperiment stress test dapat dianalisa bahwa multiple server yang memiliki load balancer modulo-based memiliki nilai rata-rata halaman yang dieksekusi lebih baik dari single server dan load balancer round robin. Dan untuk eksperimen waktu, load balancing round robin memiliki waktu respon yang lebih baik dari pada single server dan load balancing modulo-based.

2. Evaluasi Analisis Skenario Eksperiment Balancing Strategy Dari eksperiment balancing strategy dapat dianalisa bahwa load balancer round robin memiliki pembagian request server lebih baik dari pada load balancer modulo base. Sehingga dalam konsep pemetaan server load balancer round robin lebih baik daripada load balancer modulo base.

4. KESIMPULAN

Dari hasil implementasi dan analisis mengenai load balancing di internet information service dengan menggunakan algoritma round robin dan modulo based, maka dapat diambil kesimpulan sebagai berikut:

1. Pada eksperiment stress test terlihat bahwa multi web server dengan load balancer memiliki jumlah rata-rata halaman yang dieksekusi perdetik lebih banyak dibandingkan dengan single server.
2. Eksperiment stress test juga menunjukkan response time dengan single server dan load balancing dengan menggunakan round robin dan modulo based tidak jauh berbeda.
3. Penggunaan multi web server dengan load balancer mempunyai tingkat availability yang lebih baik dari pada single web server. Untuk single server, jika server mengalami down maka request tidak dapat diproses. Dan untuk multi web server dengan load balancer jika server mengalami down maka request dapat diproses web server yang masih aktif.
4. Penggunaan multi web server dengan load balancer mempunyai tingkat transparency yang lebih baik dari pada single web server. User tidak perlu mengetahui bila ada web server yang down.
5. Pemetaan server load balancer round robin lebih baik dari pada load balancer modulo based. Teknik round robin memiliki pembagian request server yang lebih merata dari pada teknik modulo-based.

5. REFERENSI

Hartono, Budi. Memahami Visual C#.Net Secara Mudah. Jogjakarta. 2008.

Junaed. Belajar Sendiri .Net Dengan Visual C#.Net 2005. Jogjakarta. 2006.

Taryana, Acep. Suswantoro, Hari. Penerapan e-learning olat dengan webserver ter Cluster untuk peningkatan kapasitas akses e-learning. Purwokerto. 2010

T. Bourke. Server Load Balancing. O'Reilly. 2001

Wahyudi, Gatot. Robust Consensus Clustering Menggunakan Algoritma Berbasis Centroid. Depok. 2011

PENGENDALIAN KUALITAS DENGAN MENGGUNAKAN METODE SPC (*STATISTICAL PROSES KONTROL*) PADA PEMBUATAN PRODUK COVER HANDLE REAR KWWF

Nurina [1)], Andaya.[2)]
[1]Teknik Industri UPI YAI, [2]Teknik Industri UPI YAI
Email: cahaya.nurl@gmail.com, andaya@gmail.com

Abstract: *Quality control is important to be done by the company so that the products produced in accordance with the standards established by the company and the standards set by the local and international agencies that manage about the standardization of quality / quality, and of course in accordance with what is expected by the average consumer specification limit Upper control and lower control limits are BKA = 0.256; BKB = 0.148; And CL = 0.202 ;. This means that production can continue to work with quality that can be controlled. Based on the analysis through the pareto diagram the rejected number of each type of disability is most dominant in the type of oily defect in the injection process. In the injection process there are 11 types of defect specifications include: silver, flow mark, oily, material left behind, stripes, scratch, ejector mark, wed line, rust, and yarn From the analysis there are 5 main factors, among others: Standards, methods, machinery and environment.*
Keywords: *Quality Control, Disability*

1. Pendahuluan

Dewasa ini perkembangan bisnis meningkat semakin ketat meskipun berada dalam kondisi perekonomian yang cenderung tidak stabil. Hal tersebut memberikan dampak terhadap persaingan bisnis yang semakin tinggi dan tajam, baik di pasar domestik maupun di pasar internasional. Setiap usaha dalam persaingan tinggi dituntut untuk selalu berkompetisi dengan perusahaan lain di dalam industri yang sejenis. Salah satu cara agar bisa memenangkan kompetisi atau paling tidak dapat bertahan di dalam kompetisi tersebut adalah dengan memberikan perhatian penuh terhadap kualitas produk yang dihasilkan oleh perusahaan sehingga bisa mengungguli produk yang dihasilkan oleh pesaing. Permasalahan kualitas telah mengarah pada taktik dan strategi perusahaan secara menyeluruh dalam rangka untuk memiliki daya saing dan bertahan terhadap persaingan global dengan produk perusahaan lain (La Hatani, 2007).

Pengendalian kualitas penting untuk dilakukan oleh perusahaan agar produk yang dihasilkan sesuai dengan standar yang telah ditetapkan perusahaan maupun standar yang telah ditetapkan oleh badan lokal dan internasional yang mengelola tentang standarisasi mutu/ kualitas, dan tentunya sesuai dengan apa yang diharapkan oleh konsumen (Vincet, 2005).

Pengendalian kualitas yang dilaksanakan dengan baik akan memberikan dampak terhadap kualitas produk yang dihasilkan oleh perusahaan. Standar kualitas meliputi bahan baku, proses produksi dan produk jadi (M.N. Nasution, 2005). Oleh karenanya, kegiatan pengendalian kualitas tersebut dapat dilakukan mulai dari bahan baku, selama proses produksi berlangsung sampai pada produk akhir dan disesuaikan dengan standar yang tetapkan

Masalah yang dihadapi oleh PT. Astra Honda Motor khususnya dibagian plastic injection yaitu pengendalian kualitas, sebab kualitas merupakan syarat mutlak dalam memproduksi suatu produk. Hambatan yang di alami yaitu sering terjadinya reject atau kecacatan, antara lain : short shot, silver, flow mark, minyak, material tertinggal, legok, belang, gores, ejector mark, wedline, karat, benang, reject awal, dan reject setting.

Akibat kecacatan tersebut maka produksi cover handle rear kwwf yang dihasilkan sering menghambat proses berikutnya. Hal ini perlu dicegah agar dalam proses assy unit tidak mengalami hambatan hanya karena komponen tersebut tidak terpasang pada sepeda motor. Selain itu juga untuk meminimalkan biaya material pada pembuatan produk cover handle rear kwwf dan tidak menggangu produktivitas.

Dengan adanya kendali mutu untuk menghasilkan produk jadi, maka bagian plastic injection dapat mempertahankan dan meningkatkan mutu dari produk cover handle rear kwwf sehingga dapat memperlancar produksi.

2. Metode Penelitian

Metode yang digunakan dalam pengendalian kualitas ini dengan menggunakan metode SPC

(Statistical Proses Kontrol) melalui penggunaan peta kendali p – Chart, peta kendali p yaitu peta kendali untuk bagian yang ditolak karena tidak sesuai dengan spesifikasi dan apakah cacat produk yang dihasilkan masih dalam batas yang disyaratkan. Selain itu peta kendali ini digunakan untuk mengukur jumlah yang di tolak berupa proporsi cacat dalam setiap melakukan observasi/mengambil sample berubah-ubah jumlahnya. Dan untuk data atribut digunakan diagram pareto, sedangkan diagram sebab-alibat atau fishbone diagram digunakan untuk mengambarkan hubungan antara masalah dengan penyebab potensialnya

3. Hasil dan Pembahasan

Agar diperoleh siklus pengambilan data yang mempunyai persentase kesalahan terkecil lebih dahulu dilakukan uji kecukupan data yang berfungsi untuk mengetahui apakah data yang diamati telah cukup atau tidak mewakili populasi dengan melihat N lebih besar dati N', maka cukup mewakili populasi. Setelah dilakukan pengolahan data dari bab sebelumnya didapatkan nilai N' sebesar 14,93 dan nilai N sebesar 20 dengan demikian data yang telah diambil dinyatakan cukup.

Dikatakan seragam apabila seluruh data yang diambil berada didaerah batas control. Dan berdasarkan pengolahan data pada bab sebelumnya diperoleh jumlah rata-rata sub group sebesar dengan banyak sub grup yaitu 4. Sedangkan standar deviasi sebesar ; standar deviasi sub grup sebesar pada pengujian yang dilakukan menggunakan tingkat keyakinan 95% dan tingkat ketelitian 5%. Kemudian menghitung batas kendali atas dan bawah, diperoleh hasilnya untuk batas kendali atas (BKA) yaitu dan batas kendali bawah (BKB) yaitu dari hasil perhitungan menunjukan harga rata-rata dari masing-masing sub grup berada dalam batas kendali dengan demikian, data telah seragam.

3.1 Analisis Perhitungan Peta Kendali p

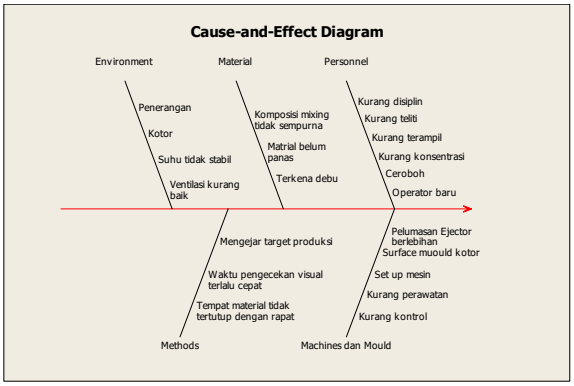
Pengendalian kualitas proses statistical (SPC) terdapat dua jenis data yaitu data variable dan data atribut. Dalam laporan kerja praktek ini data yang diambil adalah data atribut dikarenakan data menunjukan karakteristik kualitas sesuai atau tidak sesuai spesifikasi, dengan menggunakan peta kendali p maka dapat mengetahui bagian yang ditolak karena tidak sesuai, dimana dalam pengambilan sample berubah-ubah jumlahnya serta memang perusahaan melakukan inspeksi 100%.

Berdasarkan data-data yang telah dikumpulkan maka dibuat peta kendali p untuk mengetahui proses berada dalam pengendalian atau tidak. Digunakan peta kendali p karena data merupakan data atribut, dan ukuran sub grup (n) berubah-ubah yang terdiri dari 20 buah. Setelah peta kendali p dibuat terdapat sepuluh buah data yang keluar dari batas kendali yaitu pada pengukuran ke: 1,4,5,8,10,11,12,15,17, dan 18. Kesepuluh data tersebut berada diluar batas kendali karena proporsi cacatnya lebih tinggi dari batas kendali atas/BKA (UCL) sebesar dan lebih rendah dari batas kendali bawah/BKB (LCL) sebesar maka dari itu perlu dilakukan revisi agar data yang diperoleh dalam batas kendali p.

Data-data yang berada diluar batas kendali pada peta kendali sebelumnya kemudian dihilangkan/dibuang dan dibuat kembali pata kendali revisi. Setelah peta kendali p revisi dibuat dengan demikian semua data berada dalam batas peta kendali revisi. Dimana diperoleh batas kendali atas yaitu dan batas kendali bawah yaitu setelah dilakukan revisi seluruh data telah berada dalam kondisi in control statistical sehingga tidak perlu lagi dilakukan revisi lagi. Hal ini berarti produksi dapat terus dikerjakan dengan kualitas yang dapat terkendali.

3.2 Analisis Diagram Fishbone

Diagram fishbone digunakan untuk mengetahui akibat dari suatu masalah untuk selanjutnya diambil tindakan perbaikan. Diagram tersebut sebagai petunjuk factor penyebab dan karakteristik kualitas (akibat) yang disebabkan oleh factor penyebab. Dalam hal ini guna mengetahui penyebab yang mengakibatkan cacat berminyak pada Cover Handle Rear KWWF dimana cacat ini terjadi pada proses injection dapat dilihat pada gambar berikut ini:



Sumber : Hasil Pengolahan Data Dengan Minitab

Gambar 1. Diagram Fishbone Cacat Pada Proses Injection

Dengan menggunakan diagram fishbone ini maka penyebab dari terjadinya kecacatan dapat diketahui dengan sebelumnya melakukan analisis terhadap factor-faktor yang mempengaruhi antara lain seperti manusia, mesin, metode, bahan baku, dan lingkungan.

Table 1. Faktor Penyebab Terjadinya Kecacatan Cover Handle Rear KWWF

No	Faktor	Keadaan
1	Manusia	a. Kurang disiplin b. Kurang teliti c. Kurang terampil d. Kurang konsentrasi e. Ceroboh f. Operator baru
2	Bahan Baku	Komposisi mixing a. tidak sempurna Matrial belum b. panas c. Terkena debu
3	Metode	Tempat material tidak tertutup a. dengan rapat Waktu pengecekan b. visual terlalu cepat Mengejar target c. produksi
4	Mesin dan Mould	a. Kurang kontrol b. Kurang perawatan c. Set up mesin Surface muould e. kotor Pelumasan Ejector f. berlebihan
5	Lingkungan	a. Penerangan b. Kotor c. Suhu tidak stabil Ventilasi kurang d. baik

Sumber : Hasil Pengumpulan Data

3.3 Uji Kecukupan Data dan Uji Keseragaman Data

Uji variasi data terdiri dari 2 kegiatan yaitu uji kecukupan data dan uji keseragaman data. Uji kecukupan data berfungsi untuk mengetahui apakah data yang diamati telah cukup atau mewakili populasi dengan melihat nilai N' maka cukup mewakili pupolasi.Dari hasil analisis didapatkan nilai N' = 14,93; N = 20 untuk kriteria cukup atau tidaknya mewakili populasi. Kriteria ditentukan nilai N lebih besar dari pada nilai N' maka data telah cukup mewakili populasi data yang telah ada . Uji keseragaman data dari hasil analisis didapatkan nilai rata-rata sub grup = , BKA = , dan BKB = , harga rata dari masing – masing sub grup berada dalam batas kendali sehingga dengan demikian data telah seragam.

3.4 Peta Kendali p

Peta kendali p dari hasil anilisis masih terdapat banyak data yang diluar batas kendali pada pengukura ke 1,4,5,8,10,11,12,15,17, dan 18. Dimana diperoleh spesifikasi rata-rata batas kendali atas dan bawah yaitu: BKA = dan BKB = dan Setelah dilakukan revisi, maka seluruh data telah berada dalam kondisi in statistical control sehingga tidak perlu dilakukan revisi lagi. Dengan spesifikasi rata – rata batas kendali atas dan batas kendali bawah yaitu BKA = ; BKB = ; dan CL = 0.202;. Hal ini berarti produksi dapat terus dikerjakan dengan kualitas yang dapat terkendali.

3.5 Diagram Pareto

Diagram pareto untuk memusatkan perhatian pada penyebab permasalahan yang terjadi. Berdasarkan analisis melalui diagram pareto jumlah yang ditolak dari setiap jenis kecacatan paling dominan terdapat pada jenis cacat berminyak pada proses injection. Dalam proses injection terdapat 11 spesifikasi jenis cacat antara lain : silver, flow mark, berminyak, material tertinggal, legok, belang, gores, ejector mark, wed line, karat, dan benang.

3.6 Diagram Fishbone

Diagram fishbone digunakan untuk mengetahui akibat dari suatu masalah untuk selanjutnya diambil tindakan perbaikan. Diagram tersebut sebagai petunjuk faktor penyebab dan karakteristik kualitas (akibat) yang disebabkan oleh factor penyebab. Dalam hal ini guna mengetahui penyebab yang mengakibatkan cacat berminyak pada produksi cover handle rear KWWF, dimana cacat ini terjadi pada proses injection. Dari hasil analisis ada 5 faktor utama antara lain : operator, bahan baku, metode, mesin

dan lingkungan.

3.7 Langkah Perbaikan

Agar proses produksi menjadi lebih baik dan dapat mengurangi terjadinya kecacatan dalam jumlah banyak perlu diambil tindakan perbaikan dan penanggulangan terutama memfokuskan perhatian kepada operator, mesin dan mould. Maka perbaikan yang dilakukan yaitu untuk operator perlu dilakukan training agar mempunyai disiplin yang tinggi dan attitude yg baik, untuk mesin dan mould perlu dilakukan pengecekan dan pembersihan sebelum awal produksi juga dilakukan maintenance secara countinue.

4. Kesimpulan

Cover handle rear KWWF terbuat dari campuran material ABS JSR 10JK2 A (95%) dan MB BLACK 302-S (5%) setelah material mengalami proses pemanasan didalam hoper maka di injeksikan dengan mesin injection plastic ke dalam moulding.

Setelah dilakukan revisi pada peta p didapatkan total proporsi cacat : 2.032 ; dengan rata-rata sample : ; dan mempunyai proporsi cacat : 0.202

Batas-batas pengendalian yang diperoleh pada peta kendali kualitas yaitu untuk batas control/control limit (CL) : 0.202 ; batas kendali atas/uper control limit (UCL) : 0.256 ; dan batas kendali bawah/lower control limit (LCL) : 0.148

Dengan menggunakan diagram pareto didapatkan jenis reject yang paling tinggi yaitu jenis reject berminyak, yang mempunyai persentase sebesar 11.7 %. Kemudian untuk mengetahui faktor penyebab terjadinya kecacatan produk yaitu dengan menggunakan diagram fishbone dimana cacat ini terjadi karena 5 faktor utama yaitu : operator, bahan baku, metode, mesin dan lingkungan. Dengan demikian maka jenis reject ini diprioritaskan untuk proses perbaikan.

Dari hasil analisis dan pembahasan peta kendali p pada cover handle rear KWWF disimpulkan bahwa proses produksi dapat terus dikerjakan dengan kulitas yang terkendali karena spesifikasi rata-rata berada dalam kondisi in statistical control.

5. Refrensi

Hatani. La. Manajemen Pengendalian Mutu Produksi Roti melalui Pendekatan Statistical Quality Control (SQC) (Studi kasus pada perusahaan roti Rizki Kendari) Jurnal Ekonomi dan Manajemen UNHALU. 2007

M.N Nasution. Manajeen Mutu Terpadu. Ghalia. Indonesia. 2005

Vincent Gaperz. Total Quality Management. Jakarta. Pt Gramedia. 2005

